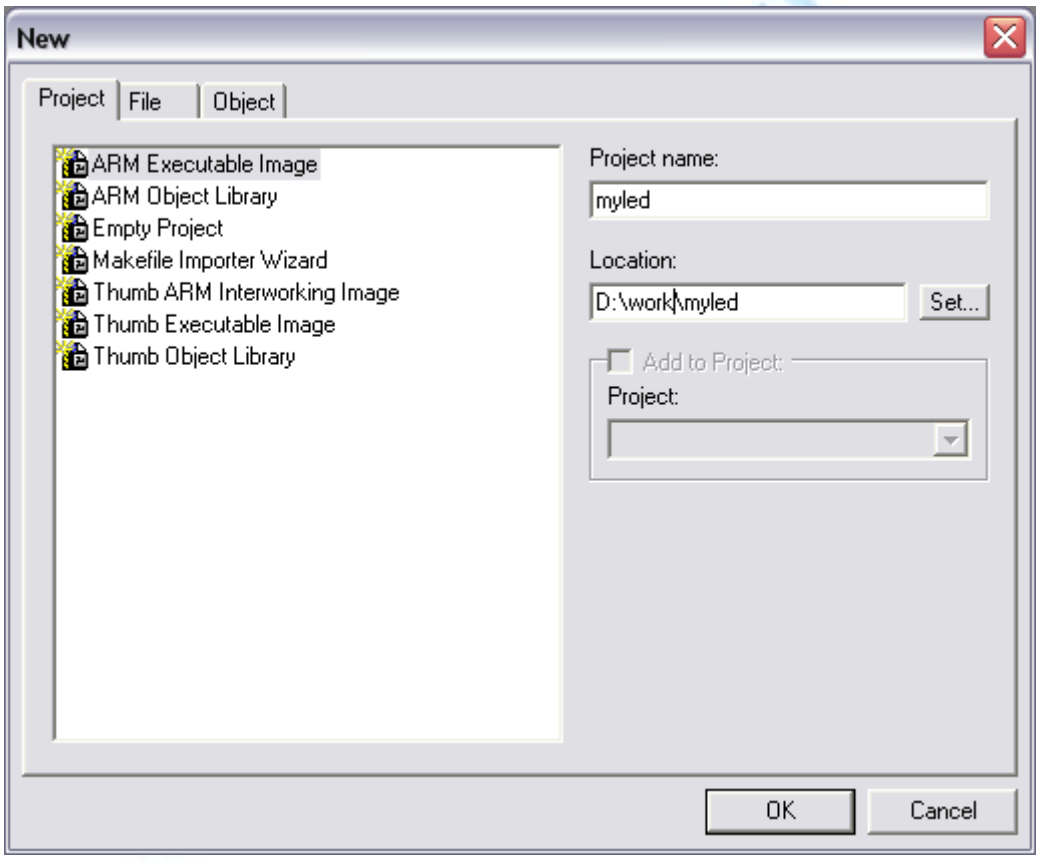
4.1 Use ADS to Create LED Project

This section with a simple concrete example, describes how to use integrated development environment ADS. Including how to create a new project, how to configure the compiler options and the compiler can generate Flash programmer to the bin executable binary file format.

4.1.1 Establishment Project

Integrated development environment in the ADS, click File > New, open a window as shown.



Can see that there are seven kinds of project types to choose:

ARM Executable Image: used by the ARM instruction code to generate ELF format image files can be implemented.

ARM Object Library: used by the ARM instruction code to generate a target file format armar library.

Empty Project: to create a database does not contain any source files or project.

Makefile Importer Wizard: used to Visual C of nmake or GNU make file into the CodeWarrior IDE project file.

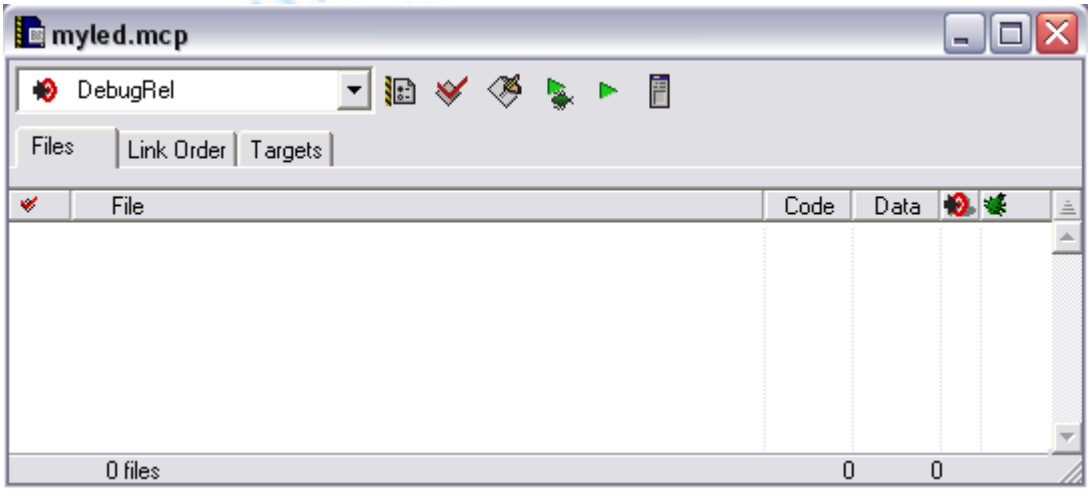
Thumb ARM Executable Image: used by the ARM and Thumb instruction mix instruction code to generate ELF executable image file format.

Thumb Executable image: for by the Thumb instruction to create an executable ELF image file format.

Thumb Object Library: for the Thumb instruction code to generate a target file format armar library.

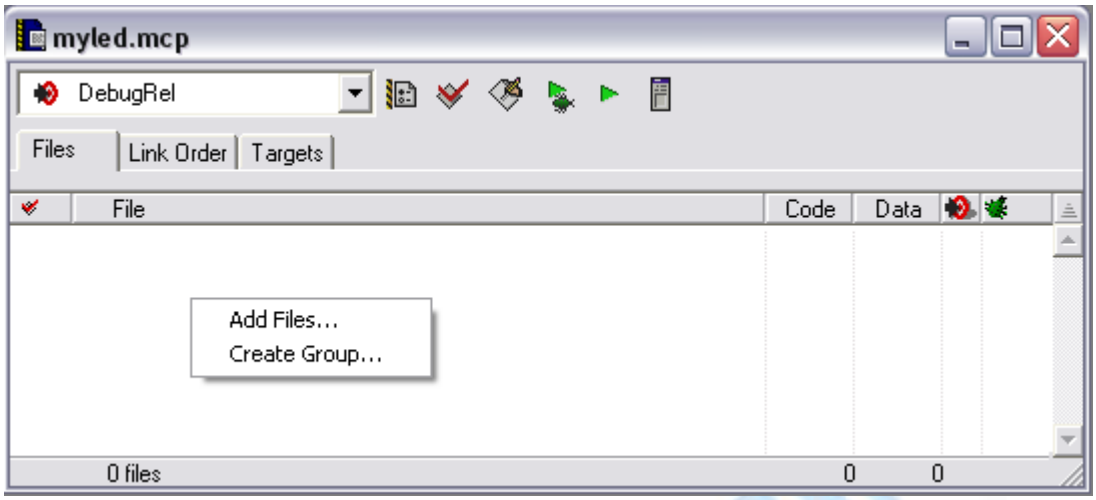
Here we select ARM Executable Image, the "Project name:" enter the project file name, in this case for "myled", click "Location:" text box "Set" button, browse to choose the project you want to save the path (the case "D:\work"), to set up, click "OK", you can create a new project called myled.

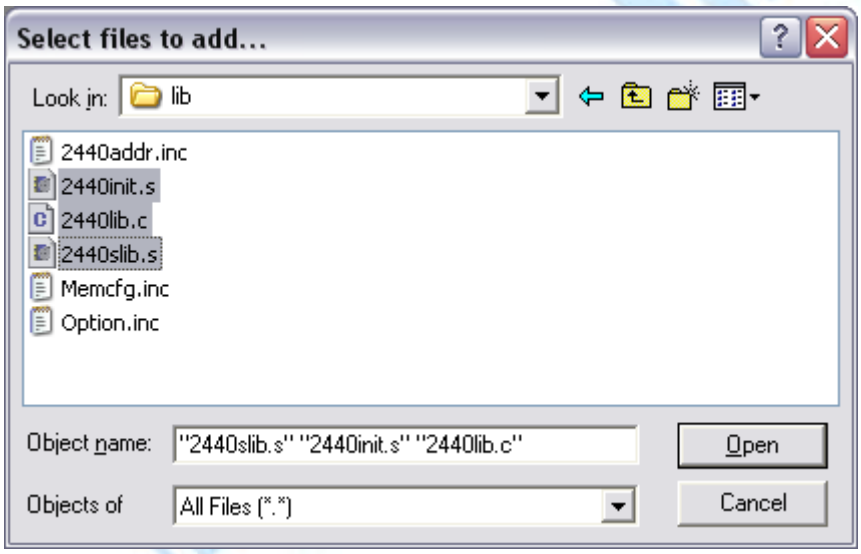
Myled.mcp this time window will appear, as shown, while in D:\work directory to create a project head recorded myled, and myled.mcp will appear in the "D:\work\myled" directory.



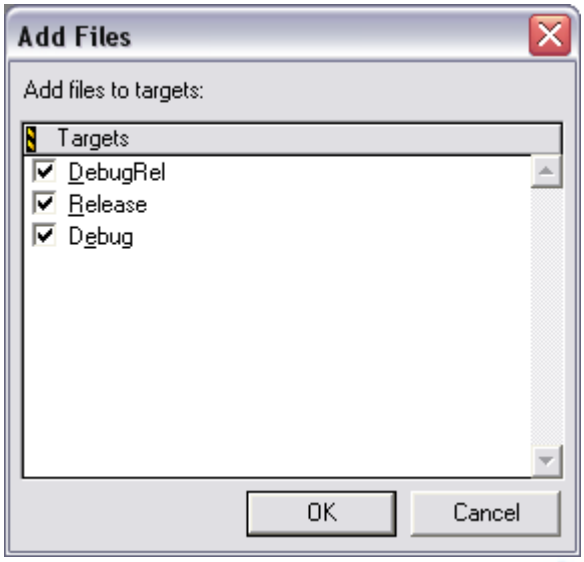
For this example, we will have been prepared and its source file directory (located in CD-ROM of "non-operating system sample code \myled "folder) to copy the project directory myled, figure.

And then project myled.mcp window, point the right mouse button or the ADS menu Project > Add Files ..., Figure began to claim the need to add the project's source code.





Click "Open" button to determine, at this time will be shown out of the following tips to choose the window.



Please note here, we create a new project, ADS default target when DebugRel, in addition to two available targets, respectively Release and Debug, meaning they are as follows:

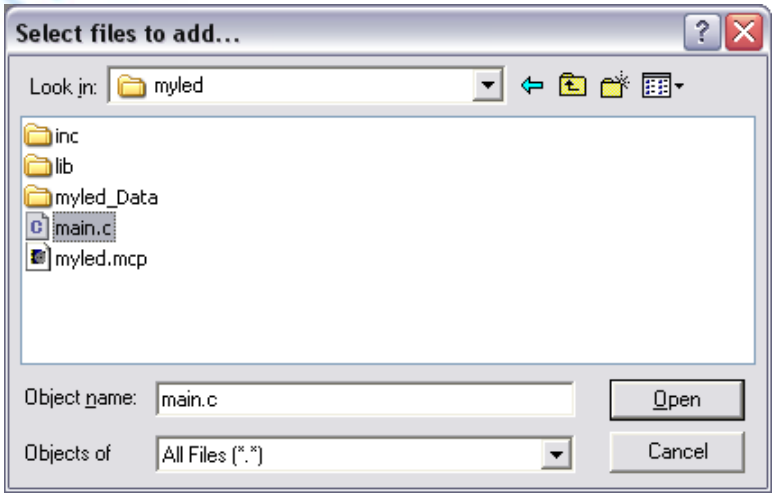
DebugRel: the use of the target option to generate the target in time for each source files to generate debugging information.

Debug: Use the target option to generate goals in time, will be for each to generate the most complete source code debugging letter income.

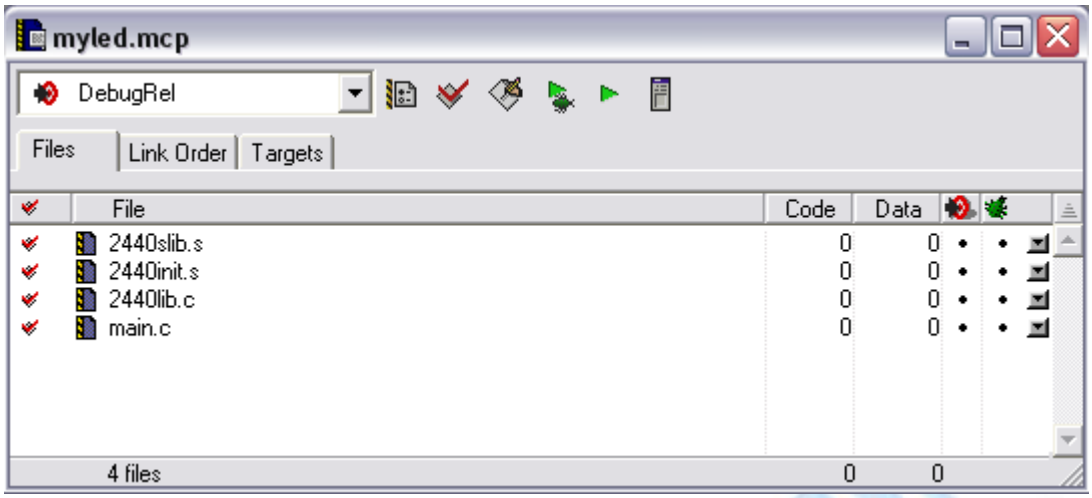
Release: Use the target option to generate goals in time, will not generate any debugging information.

In this case, we use the default options DebugRel.

Then joined main.c to myled.mcp projects:



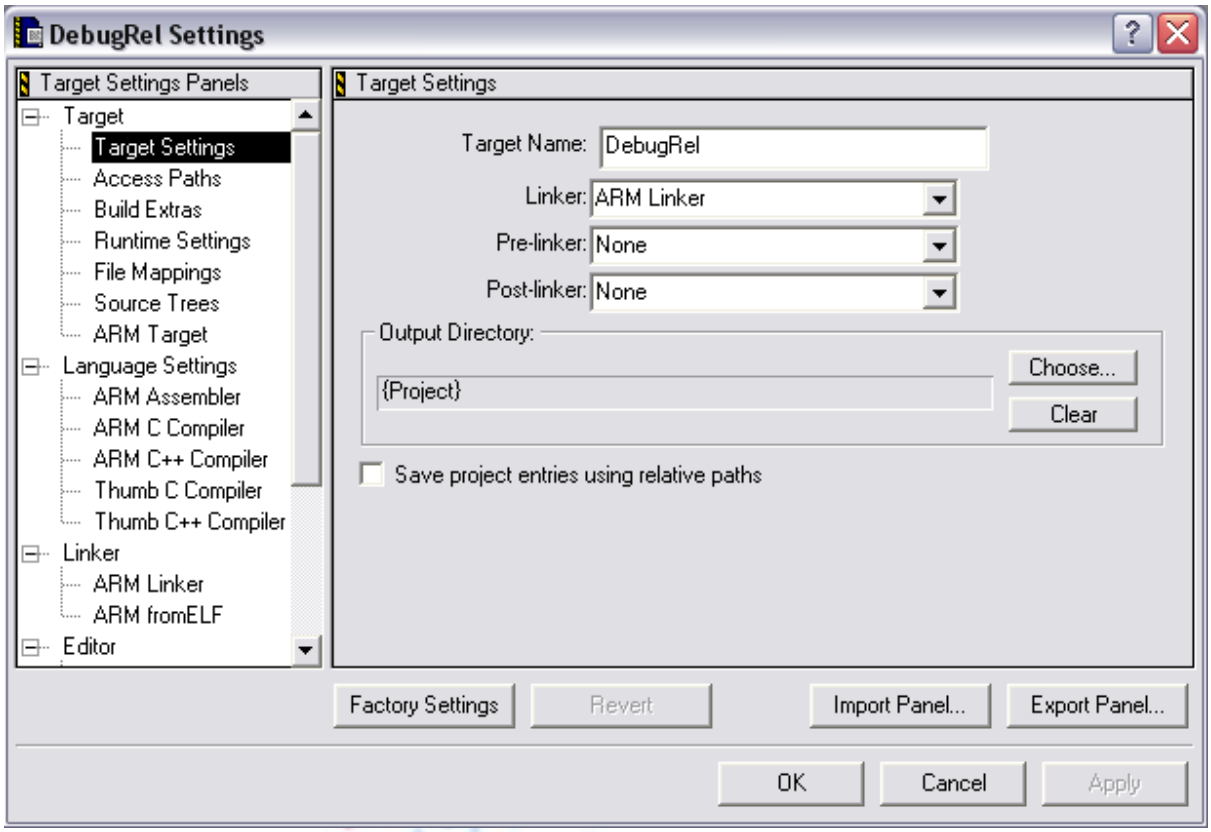
So far, a complete project has been established, following the work we have begun to compile and link configuration.



Enable Your Design
ThaiEasyElec.com
On-line Electronics Shop for Embedded System

4.1.2 Compile and Link Project

Compile and link in before, first of all need to be configured to generate the goals, points, Edit menu, select "DebugRel setting ... "(Note: this option because of the different users to choose different goals), the settings appear as shown in figure.



There are many settings here; we introduced some of the most commonly used options.

Target Setting

Target Name text box displays the current settings of the target.

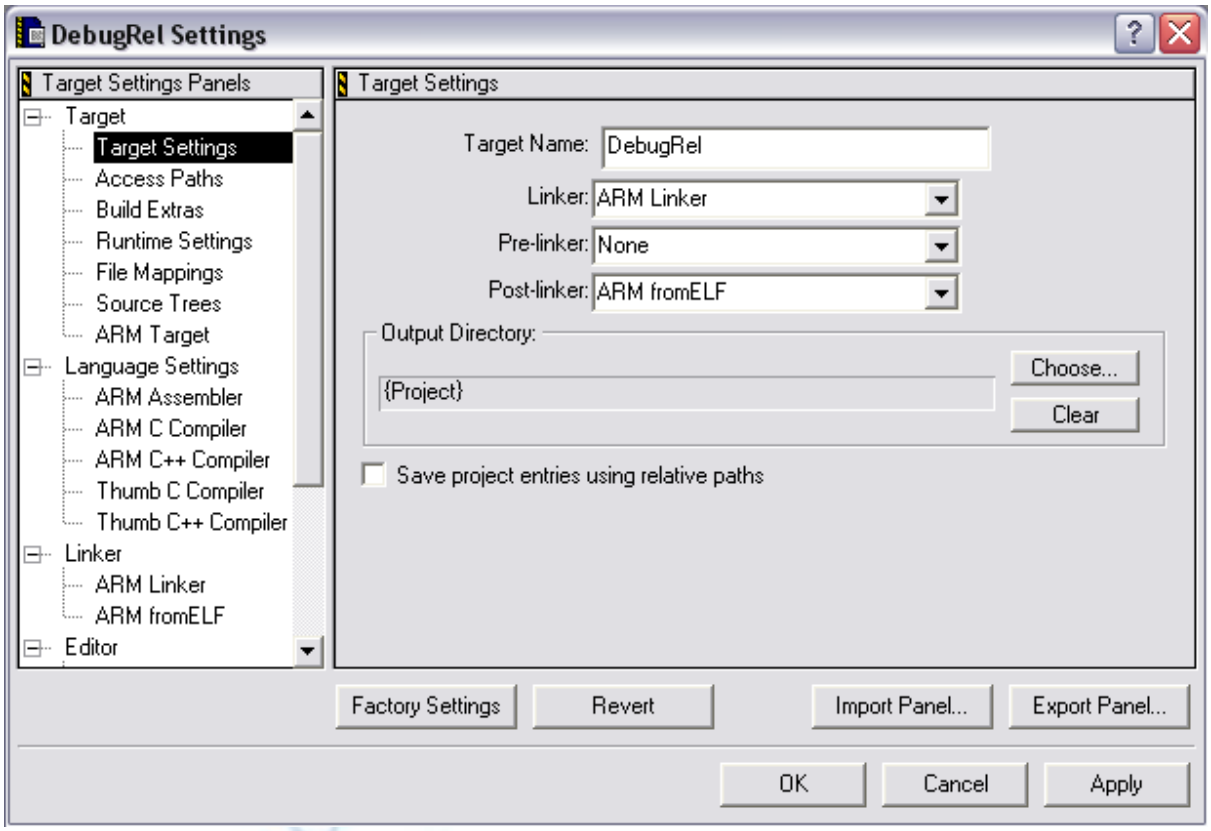
Linker options for users to use even the linker, where the default choice of ARM Linker, the use of the Linker to link the use of armlink compiler and assembler to generate the corresponding object file.

In Linker settings, there are two options, None is not representative of all the source code to generate the target document chain then, ARM Librarian that will be compiled, or compilation of the target file is converted to ARM library file, for this example, so that Linker to use the default ARM Linker.

Pre-Linker: At present, ADS does not support this option.

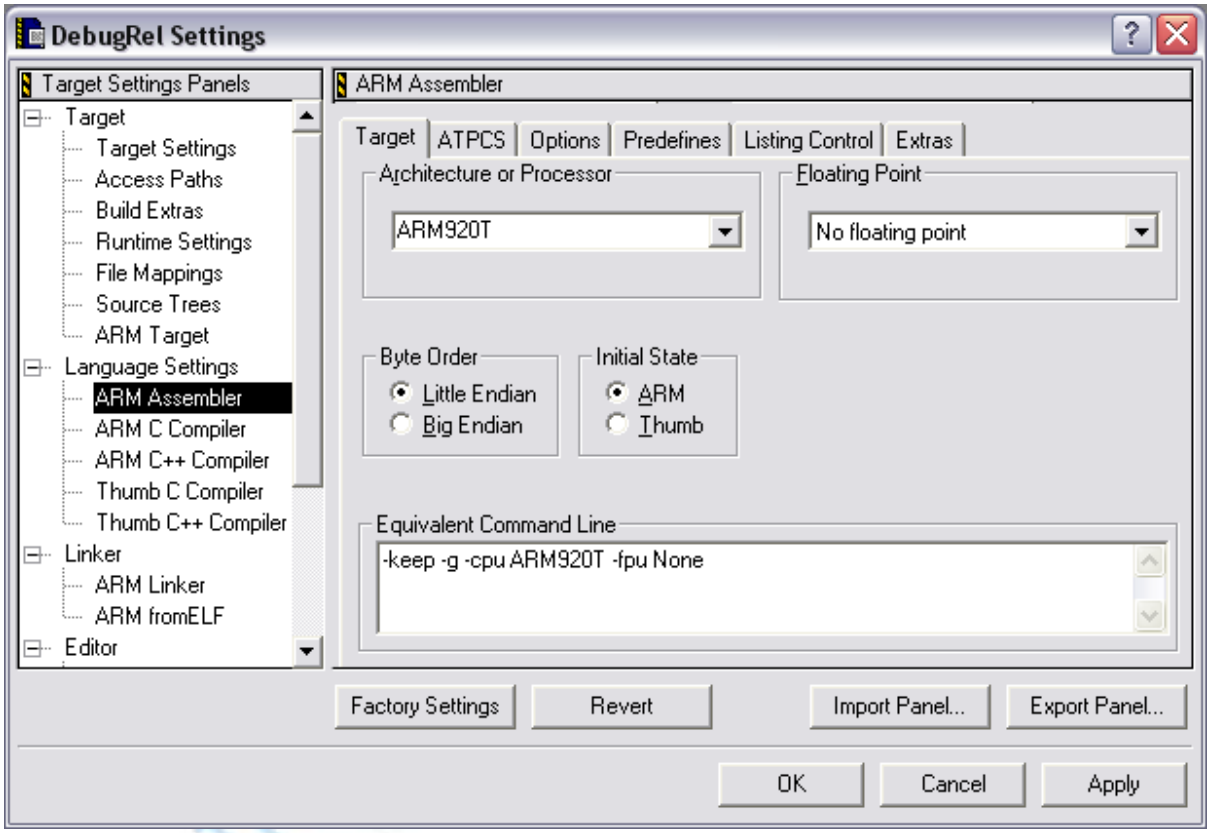
Post-Linker: the link after the completion of choice, but also on the operation of the output file. Because in this case, I hope Health as a programmer in to the Flash binary code, so this option ARM fromELF, generated in the link Image file and then call the command fromELF containing debugging information in the ELF format image file is converted to other file formats.

Target Setting to choose the final set as shown:

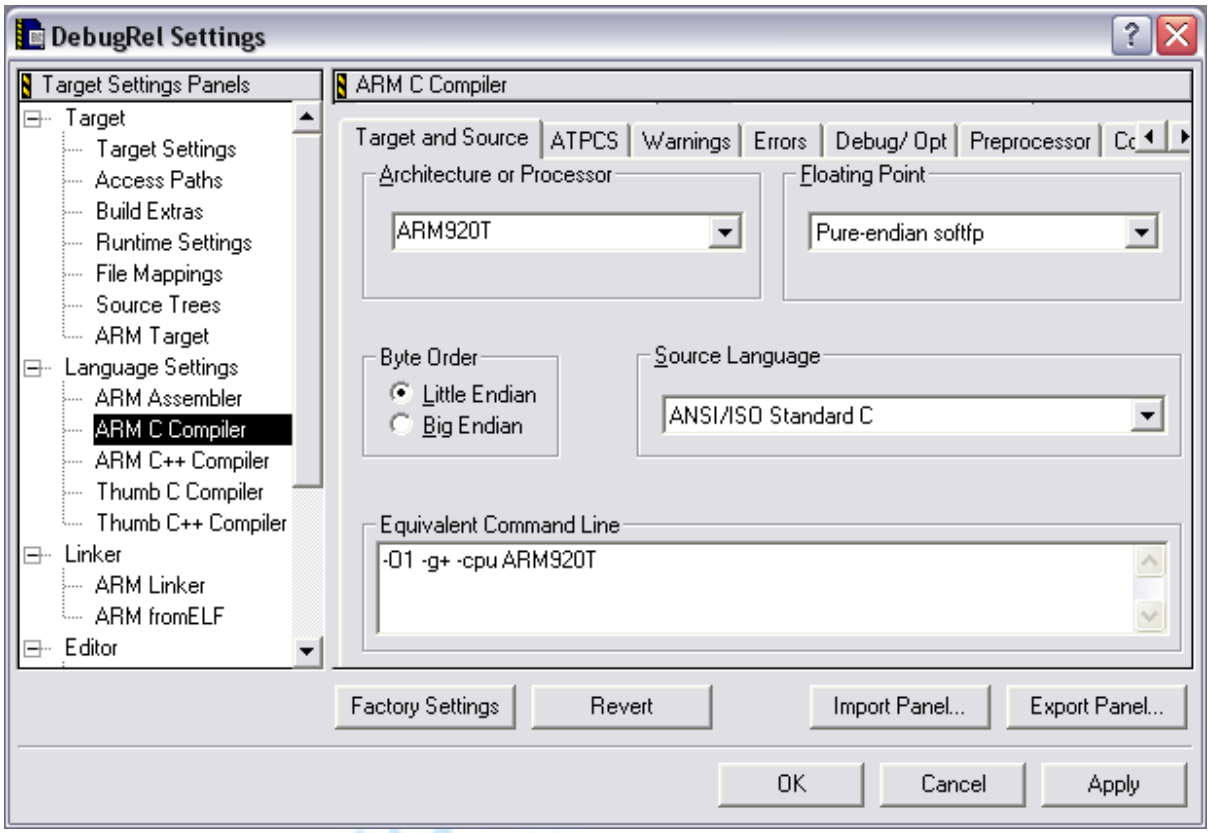


Language Settings

Because in this case contains the assembly code, all used to the compilation, and point of access to choose ARM Assembler, on the right a corresponding set of options, in the ADS with the Integrated Development Environment is a compilation of armasm, the default node ARM system conformation is the ARM7TDMI, in this proposal to change the ARM920T, the default byte order is little-mode, and other settings, the use of default value can, as shown:



This case also includes a C language code, it also need to set up the option ARM C Compiler, ARM-point access options C Compiler, in the right side of a corresponding set of options, in the ADS with the Integrated Development Environment is a compilation of armcc, silent identification of the ARM architecture is the ARM7TDMI, in this proposal to change the ARM920T, the default byte order is little-model He set the default value can be used, as shown:



Careful readers may notice, in the lower right foot box set, when a set of modified, the line An option the corresponding changes will occur, in fact, this line is shown in the corresponding compiler option or link, because of CodeWarrior, developers do not have to go obstetric variety of command-line options, as long as the interface Select or Revocation of an option, the software will automatically generate the appropriate code, the order for the habit in the DOS box, type command-line users provides a great convenience.

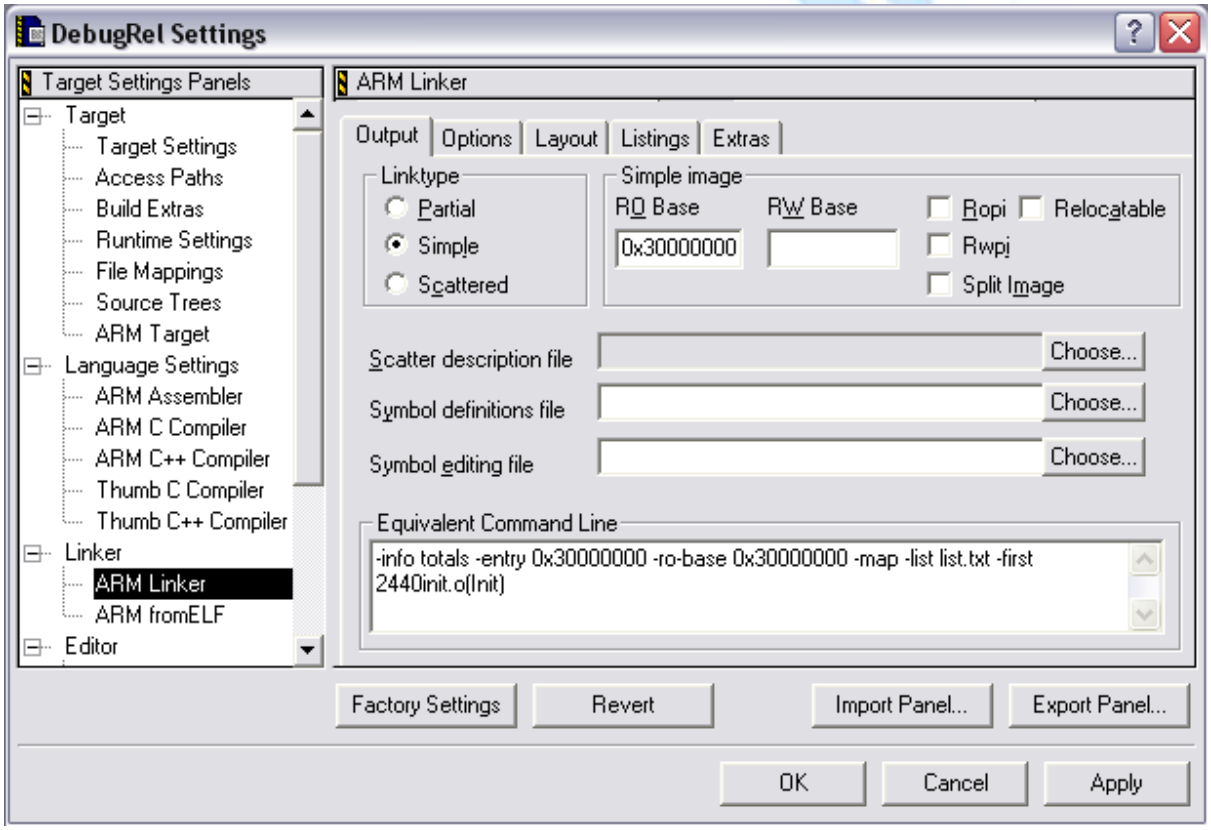
Linker Set

Tip: If you are not familiar with the ADS set, you can use our default project a document, the following sections is for reference only. Access points to choose ARM Linker, in the right

side of a corresponding set of options, we have introduced in detail the settings box, Because these options to generate the final document has a direct impact.

Output in the label, Linktype link is provided in three ways. Partial means that only the Department of linker sub-link, after part of the link destination file generated can be used as a further link after the input file. Simple square Link type is the default mode, but also links to the most frequently used methods, it generates a simple link to the goal of ELF format text Pieces, using a linker to specify the way the address mapping. Scattered linker made the way to format text according to scatter items designated address mapping, to generate the complexity of the ELF format image files, this option is rarely used in general.

(-info totals -entry 0x30000000 -ro-base 0x30000000 -map -list list.txt -first 2440init.o(Init))



In this case, we have chosen to use the Simple way, here are some settings:

RO Base: This box set contains the text of the paragraph add-RO domain and run the domain for the same address, the default is 0x8000. Here the user according to their own hardware to the actual SDRAM address space to change the address, to ensure that fill in the

address here, running, SDRAM address space beyond the scope of arrival, in view of the target board, SDRAM is the spatial extent 0x3000000-0x34000000, so here is set to 0x30000000.

RW Base: This box set contains the text of RW and ZI output section is running the domain address. Split if the selected option, Linker-generated image file will contain two to load the domain-domain and two runs, this time by RW Base in the address input contains the RW and ZI output section of the domain set up and run to load the domain-domain address.

Ropi: select this setting will tell the link output allows RO paragraph contains the domain location of the operation has nothing to do. The use of the option, the linker will ensure that the following steps: time to check the validity of the re-addressing; to ensure that any self-armlink Generated code is read-only position-independent.

Rwpi: select this option allows the link will tell RW and ZI output includes the operation of the domain has nothing to do above. If items have not been selected, the domain on the logo for the absolute. Each input can be written and read and write above position must be unrelated. If this option is selected, the link will carry out the following steps:

Inspection read / write operation of the properties of the input domain is set up above the location of irrelevant attributes;

Check in between the paragraphs re-addresses the validity of;

In the Region$$Table and ZISection$$Table add sb SRAM-based options.

RW Base option requires the value, if there is no value specified, the default is 0.

Split Image: Select this option to include RO and RW output of paragraph 2 of the load domain is divided into the loading domain, a RO output paragraph is included in the domain, and one contains the RW output section of the domain.

RW Base option requires this value, if not given the value of RW Base option, the default is 0.

Relocatable: Select this option to retain the image file offset of the re-addressing. Offset procedures such as loading provides useful information. Option in the Options, it is necessary to

arouse the reader's Image entry point is the text box. Image file that specifies the entry point address of the initial value, when the image files are loaded, the loader will jump to the Address Branch. If needed, users can enter in the text box below the entry point format:

Entry point address: This is a value, for example-entry 0x0

Symbol: The image file option specifies the entry point for the symbol represented by the address, such as:-entry int_handler, if the symbol definition of the existence of multiple, armlink will produce an error message.

offset + object (section): the option to specify the target file in a particular section of the offset within the Department for an image document entrance address, such as:-entry8+startup (startuoseg) in the designated entry points here to set ELF image text pieces of the entrance to the address. Require attention is the main symbol cannot be applied here as a symbolic entry point address, it will be similar "Image does not have an entry point (Not specified or not set due to multiple choice)" the error message. Layout options in the need to set goals asm.o the init file for the entry point of the entire document. Regarding ARM Linker settings are still many who want to further in-depth understanding of the reader, you can view the help file, there is a detailed introduction.

There is another in the Linker under ARM fromELF:

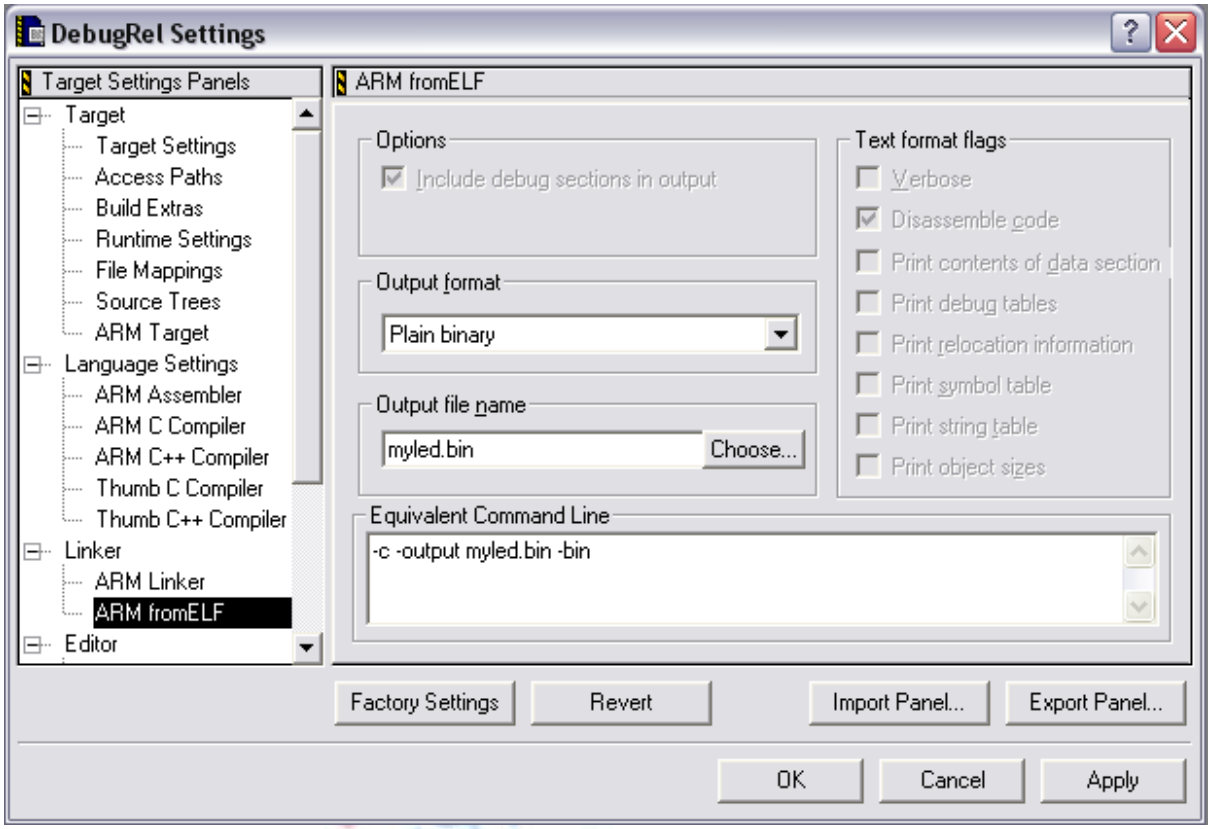
fromELF is a practical tool to achieve the linker, compiler and assembler code output format to for the function. For example, an executable ELF image file formats can be converted to a ROM programmer's binary format; Output file for an anti-compilation in order to extract the target file size, symbols, and the string table as well as re-addressing, such as letter Income. Target set up only in selected Post-linker, before payment can use this option.

In the Output format drop-down box, to provide users with a wide range of the target format can be converted, in this case select Plain binary, this is a binary executable file format can be burned target board wrote in the Flash.

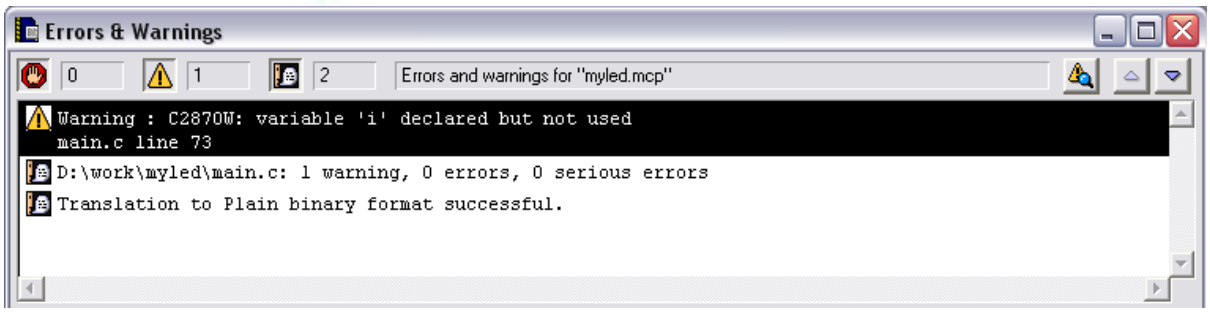
In the Output file name text field enter the expectations of the output files generated by the path, or by clicking the Choose ... button from the file dialog box, select the output file. If the text in this field do not enter a path name, then generate the binary file stored in the project

directory is located. Related to these settings, after the make of the works, the CodeWarrior IDE will call in the link after the completion of fromELF to deal with the generated image file.

For the purposes of this example project, this is completed prior to make the settings work. (-c -output myled.bin -bin)



Click on the CodeWarrior IDE menu make menu under the Project, the project will be able to compile and link the. Link the results compiled in Figure:


Errors & Warnings
0
1
2
Errors and warnings for "myled.mcp"
Warning : C2870W: variable 'i' declared but not used
main.c line 73
D:\work\myled\main.c: 1 warning, 0 errors, 0 serious errors
Translation to Plain binary format successful.

Finally, in "D:\work\myled\myled_Data\DebugRel" directory generated myled.bin, as well as myled.axf document, which is used for debugging, and Figure: